EMBRACE THE

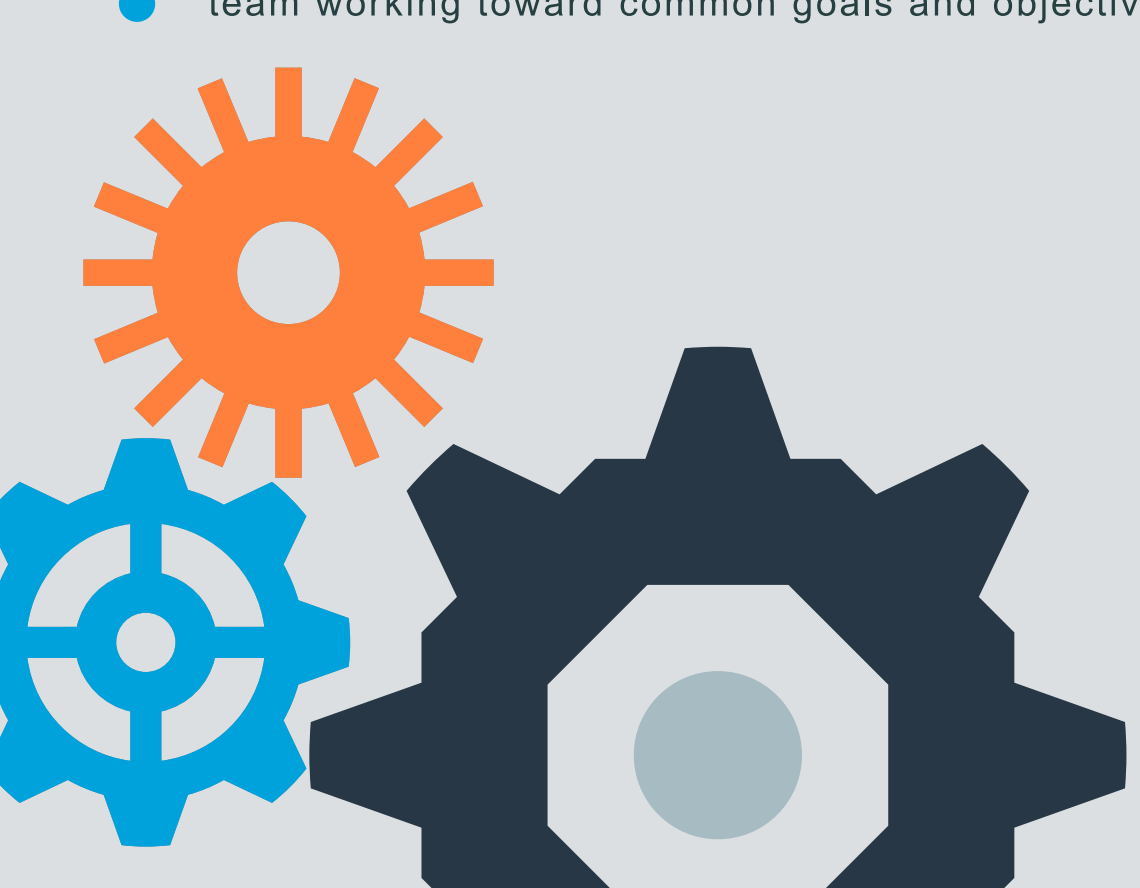
DevOps Lifecycle

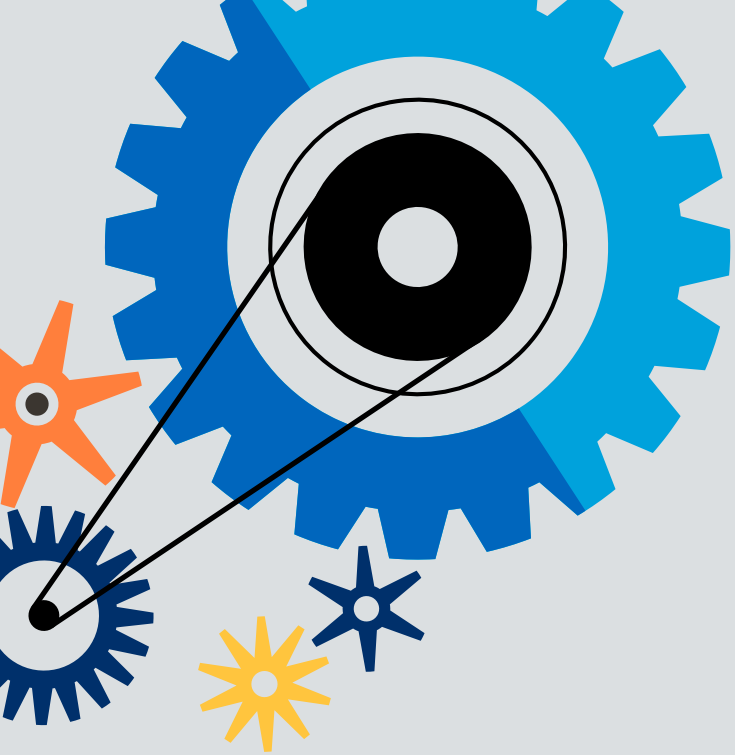
Stop me if you've heard this one before...

DevOps is about culture.

To be fair it seems like just about everything in business these days is attributed to culture. The important question is, "What exactly do we mean when we say 'culture'?"

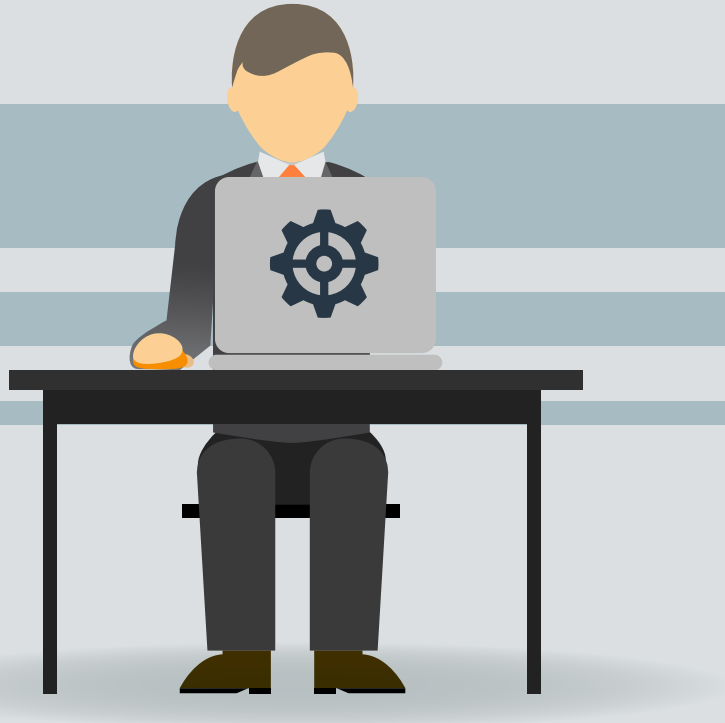
On the one hand, the focus on culture is about a shift from the traditional separation of duties where developers develop and operations operates. These two IT functions have inherently conflicting objectives—developers introduce as much change as possible and operations try to minimize change in order to deliver IT resources reliably. DevOps is about changing the mindset so that everyone is on the same team working toward common goals and objectives.





The other reason culture is stressed so much when talking about DevOps is that culture is more important than the tools or software solutions used to support IT processes. There are definitely tools that facilitate various aspects of a DevOps environment, but DevOps isn't just about the tools. A fool with a tool is still a fool. You have to make sure the tools are being used in the right ways and for the right reasons.

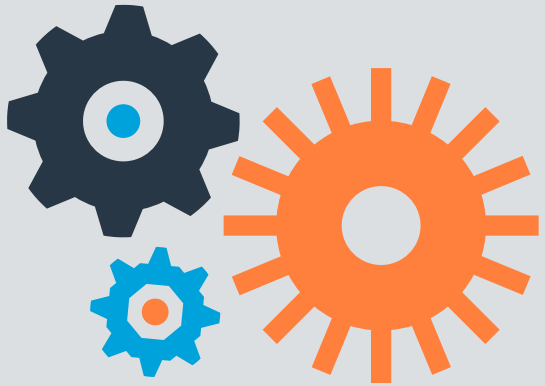
That said, technology and associated software plays a significant role and an effective DevOps environment. DevOps is about continuous delivery while maintaining quality, and this is only possible, if the right tools are chosen to support relevant people and processes. This means that it's essential that the tools and applications used to develop, test, deploy, monitor, and manage applications all work seamlessly together.



Keep CALMS and DevOps

The importance of culture is reflected in the fact that it's the first phase of the DevOps lifecycle, summed up by C.A.L.M.S. framework for DevOps:

- Culture
- Automation
- Lean
- Measurement
- Sharing



"To define culture and what it means for DevOps you have to understand the starting point, or the challenges that your enterprise's operating model currently faces in regards to software strategy and what the end, new 'DevOps' operating model looks like."

explained

John Rakowski

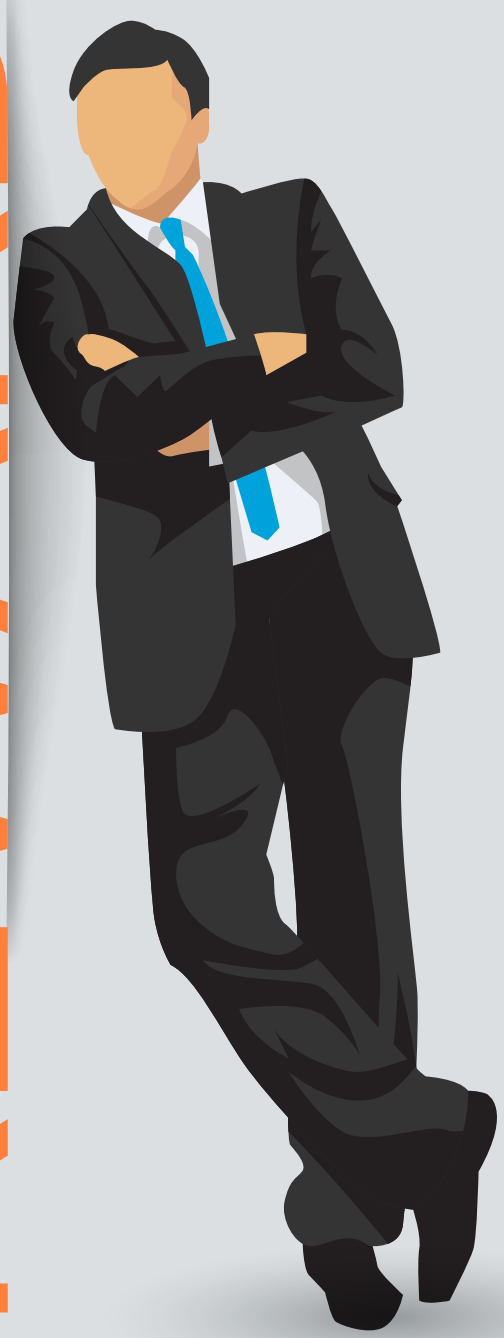
director of product marketing for Application Performance Management (APM) and analytics at AppDynamics, in a blog post.

The most common and obvious culture change is removing the segregation between organizational silos and moving to a collaborative model where teams and individuals work together to achieve better results faster. The traditional departmental roles introduce unnecessary bureaucracy and result in bottlenecks and finger pointing. DevOps empowers teams and individuals to take responsibility and function more efficiently.

There are other cultural shifts that need to occur for many organizations as well. Shifting from a fear of failure to a culture that rewards initiative and innovation even when it doesn't work. Shifting from a focus on technology to an obsession with customer experience and satisfaction. Shifting from mountains of meaningless data to business insight and actionable intelligence.

It all starts with culture, but there are DevOps tools available for each stage of the DevOps lifecycle. The first step—after culture—though, is building the right foundation.

REWARD



Foundational Tools

There are a variety of approaches to DevOps and a plethora of tools available for different elements and stages of DevOps, but all of them start from or build on the same premise: virtualization and cloud infrastructure combined with application intelligence.

Technically it's possible for an organization to adopt a DevOps culture without moving to the cloud or embracing virtualization, but it wouldn't be very effective in the long run. Once you get the DevOps culture in place, the actual practice of DevOps is about automation and agility—two things that rely heavily on the ability to have software manage aspects of the network and server infrastructure. You can't programmatically adapt to fluctuating demand using physical servers in a local data center. Even in a local datacenter virtualization is a crucial element of DevOps automation.

Virtualization enables you to implement and manage virtual hardware as a construct of the software. This is because a virtual server, virtual router, and other virtual components are themselves also just code being executed within a hypervisor platform, entire network infrastructures can be created at the push of a button.

Virtualization

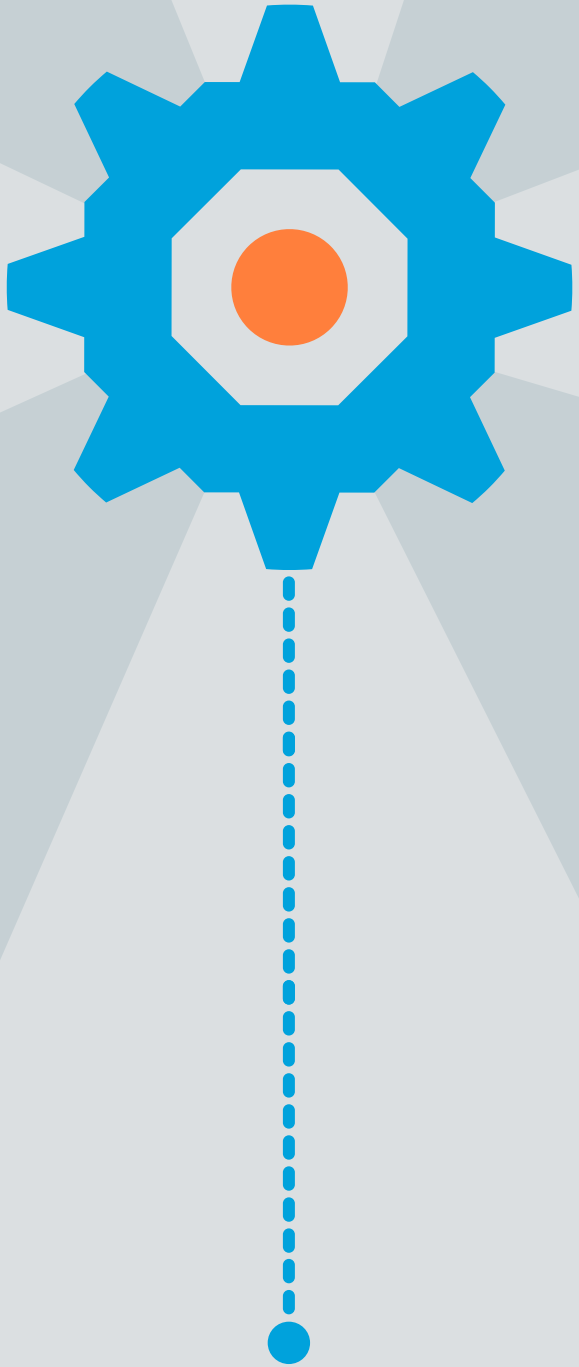
VMware
Xen
VirtualBox
Hyper-V

Cloud / Infrastructure

Azure
AWS
Rackspace
SoftLayer
Joyent
Cloud Foundry

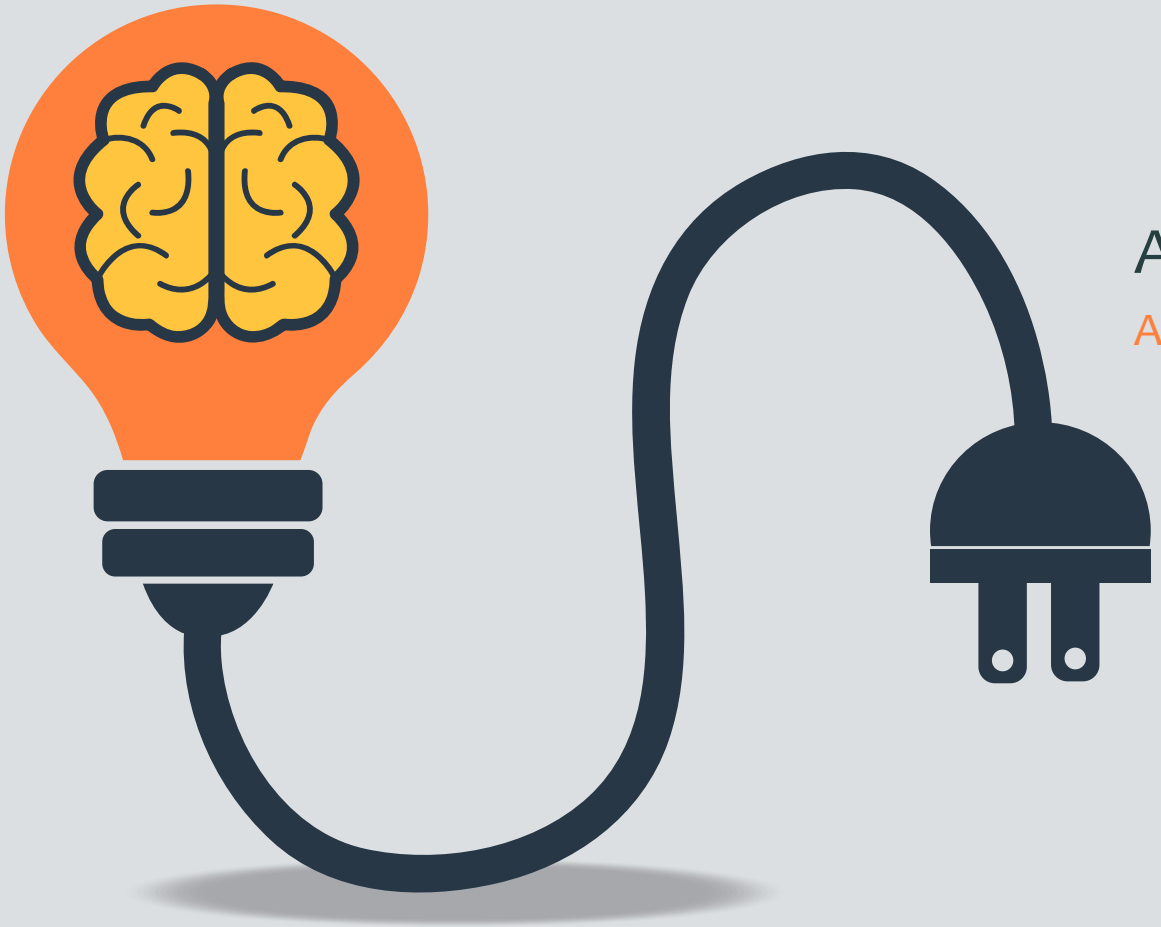
As demand exceeds the capacity of the available server, new virtual servers can be created automatically and instantly to resolve the problem. When the surge in demand subsides the additional virtual resources can simply be shut down. Virtual servers and virtual infrastructure give you the flexibility to scale dynamically, and are much more efficient and cost-effective than trying to accomplish the same thing with traditional hardware.

You can take advantage of virtualization within an on-premises datacenter on your own hardware, or in a private or public cloud. The benefits of using a cloud platform like SoftLayer or Azure is that you only pay for the resources you use in a given month. The expense of acquiring, implementing, and maintaining the underlying hardware falls on the cloud provider—enabling you to focus on productivity and customer experience rather than worrying about the hardware infrastructure. Major cloud providers also have redundant datacenters in geographically separate regions to provide resiliency and ensure you have access to your applications and data even if an outage or natural disaster impacts a given site.



The third element of the DevOps foundation is application intelligence. To implement, manage and maintain a DevOps environment effectively, executive leadership, developers, IT operations and other stakeholders need to be able to measure and monitor application performance. The ability for different teams and individuals to collaborate in one console and have access to relevant application intelligence in order to optimize software strategy is fundamental to DevOps.

The cloud and virtualization form the foundation of a DevOps framework. To succeed at DevOps organizations must also have access to real-time app-centric information that can be displayed in context of different audiences within the business to monitor performance and provide relevant application intelligence. Once you embrace DevOps culture, these are the table stakes you need to have in place before you can begin to consider and implement the other tools that comprise DevOps.



Application Intelligence

AppDynamics

DevOps Lifecycle

With the DevOps culture and a solid foundation of cloud and virtualization in place you're ready to roll. Rather than choosing DevOps tools by sifting through a laundry list of popular services and applications, though, it's better to consider the DevOps lifecycle and approach the issue of selecting tools based on the goals and objectives of the different stages. There are some tools that provide broad features and capabilities that cross over from one stage to another of the DevOps lifecycle, but in general there is a logical flow to developing and deploying software with DevOps and there are tools uniquely suited to help you automate and manage that flow.

Let's take a look at the DevOps lifecycle stages and the DevOps tools available for each. Keep in mind that DevOps isn't a linear progression, though, so some of the stages overlap or feed each other in a perpetually cyclical way:

Plan

The first step in developing a new app or updating an existing app is to have a plan. Application analytics of existing apps can help developers plan new apps and features more effectively. Developers need to have an understanding both of the business objectives of the project as well as the technical benefits users expect and the infrastructure limitations that might impact app availability and performance.

The first step in planning is an extension of the cloud and virtualization foundation we discussed earlier. You have to build a framework on that foundation and the framework consists of the Web servers, databases, and search tools the rest of the apps will be built on or integrated with. You have to weigh the pros and cons of different options to choose the tools that work best for you. Keep in mind that other tools and applications you run will need to be compatible with or integrate with your Web server and database choice so there could be a cascade effect that impacts other decisions.

Web Servers

NGINX
Apache
Azure / Windows Server

Databases

MongoDB
Cassandra
hBase
MySQL
PostgreSQL
Redis

Search

Solr

DevOps Tools

Mule ESB, F5, Amazon AWS extensions
Apica
ActiveMQ
RabbitMQ
memcache
squid

Design / Architect

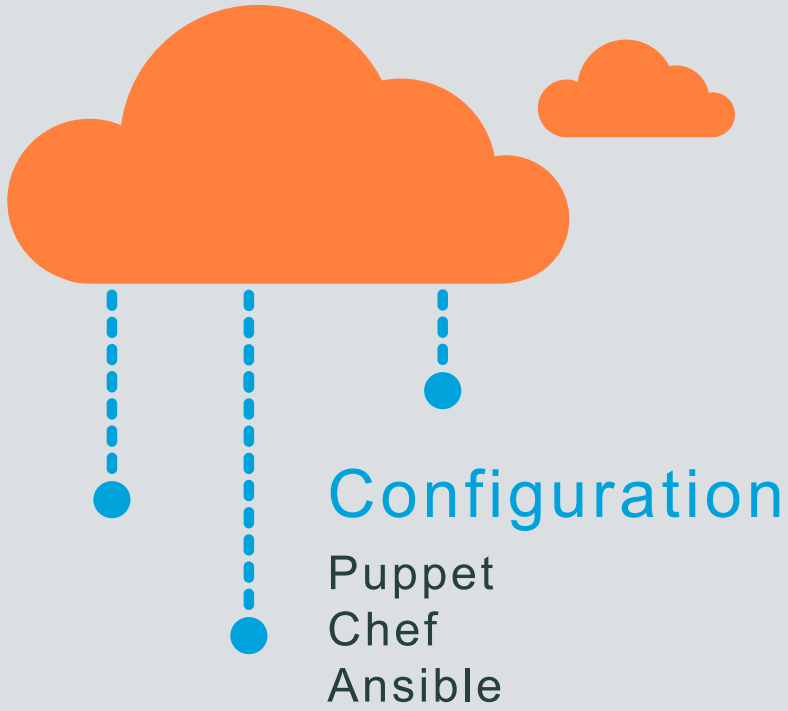
One of the mistakes organizations and developers often make is to design applications in a vacuum. Creating a checklist of features and capabilities and creating an app that provides them is a waste of time if the customer or end users don't want those features and won't use those capabilities. A DevOps approach to design and architecture flips that premise upside down to make sure the design is customer-centric in the first place.

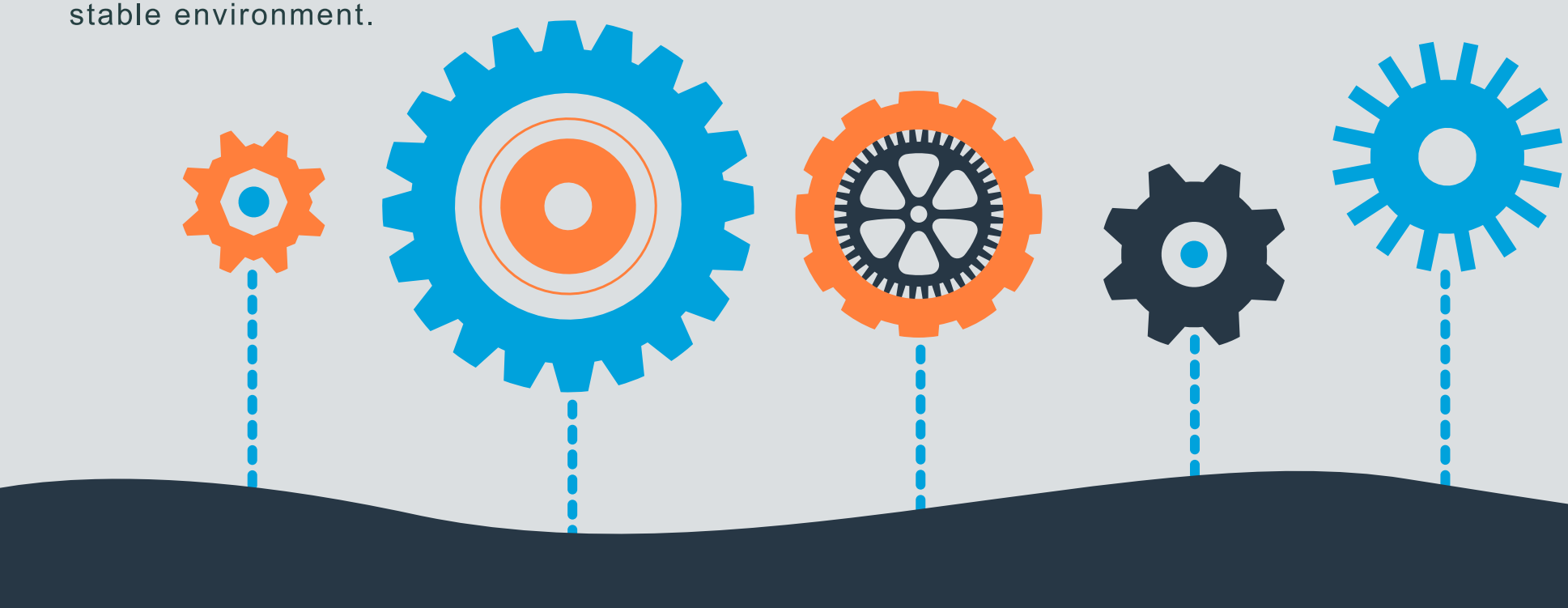
There are a variety of tools and methods available to help out with this phase of DevOps development. Real-time user monitoring and analytics from existing apps are very effective for determining what customers or end-users want. It may be that a given feature or capability just isn't necessary, or perhaps it's just not used because it's too complicated or there are performance issues that prevent it from working properly. You can monitor which features and functions get used most frequently and which don't get used at all, and you can analyze usage and performance to identify potential issues.

Implement / Deploy

This is where you start getting down to business—and where a lot of the automation part of the DevOps CALMS framework comes into play. Configuration tools, container platforms, and automated testing have completely changed the development landscape and enabled organizations to take full advantage of the cloud and virtualized foundation.

One of the challenges of a virtualized cloud environment is that it can be ethereal. It can completely change at the push of a button. The fluid and dynamic nature of a DevOps environment is one of its strengths, but it also presents unique challenges for IT when it comes to maintaining stable access to network resources and ensuring that the IT infrastructure is secure. Configuration tools like Puppet, Chef, and Ansible enable organizations to manage IT configuration as modular components and automate implementation to ensure a consistent, reliable, stable environment.



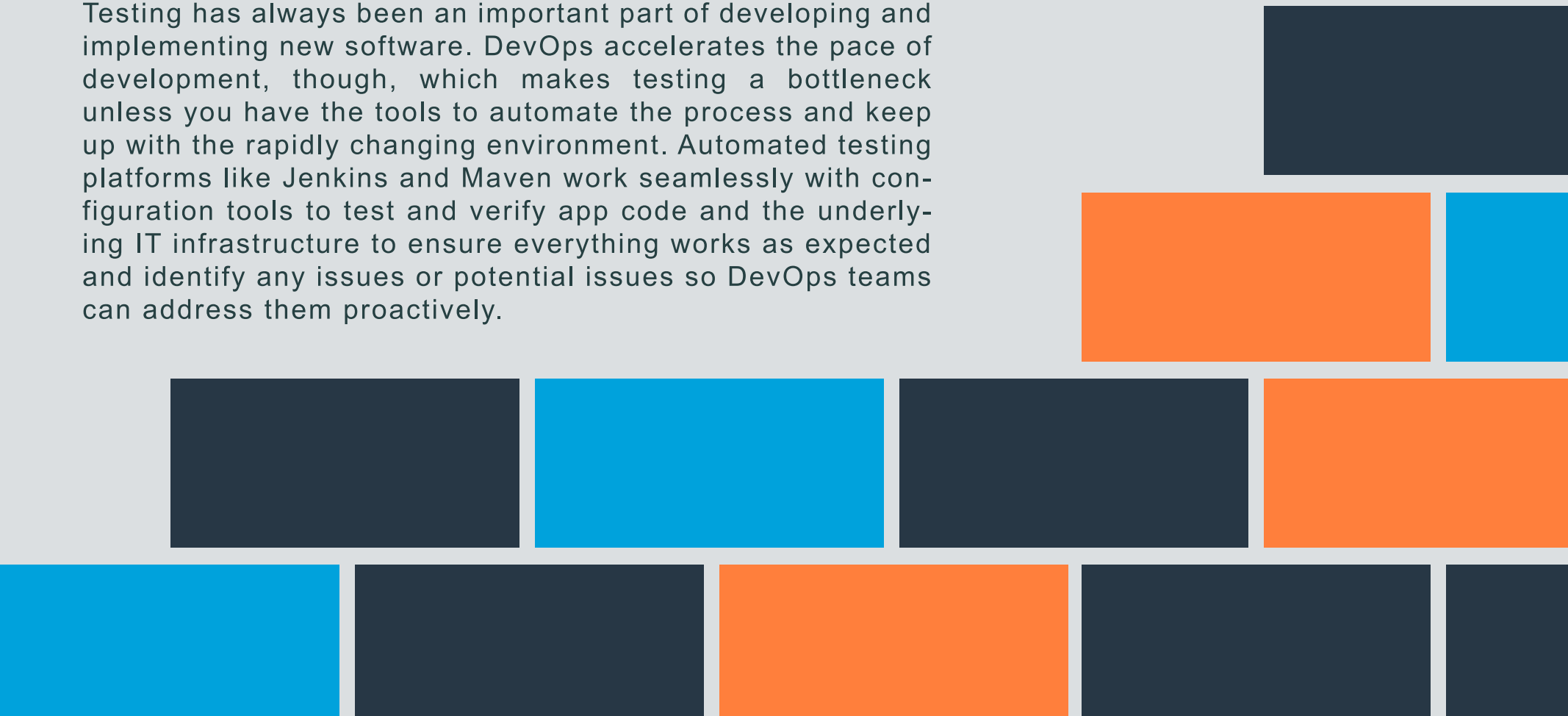


Containers take the modular approach to development a step farther. A container is an abstraction that separates the application itself from the underlying operating system—allowing a containerized app to be deployed and re-deployed in any cloud or server environment that supports the container platform. Container technology has become so integral to DevOps and app development that rival companies and major tech partners joined forces to form the Open Container Project to ensure that containers comply with common standards and prevent organizations from getting painted into a proprietary corner.

Testing has always been an important part of developing and implementing new software. DevOps accelerates the pace of development, though, which makes testing a bottleneck unless you have the tools to automate the process and keep up with the rapidly changing environment. Automated testing platforms like Jenkins and Maven work seamlessly with configuration tools to test and verify app code and the underlying IT infrastructure to ensure everything works as expected and identify any issues or potential issues so DevOps teams can address them proactively.

Testing

Jenkins
Ant
Maven



Maintain / Run

You're not done just because an app is developed and implemented. Once the app is being actively used you have to monitor performance and maintain it as problems arise or end-user report issues. Ideally, DevOps teams should identify and resolve issues before users notice or report them, which requires logging and alerting capabilities.

The dynamic nature of DevOps makes it more crucial than ever for organizations to automate logging and alerting efforts. Apps and the underlying IT infrastructure can change frequently so an app that was performing optimally one day may be dysfunctional the next. The configuration management and automated testing discussed earlier should alleviate most issues before they happen, but problems will still inevitably arise.

Logging tools gather data related to relevant metrics. Collecting data doesn't solve anything in and of itself, though. In fact, just collecting data without context can create more problems than it solves. A good logging tool will also correlate the data to produce actionable information DevOps teams can use.

Alerting tools help DevOps teams to be aware of issues in real-time and address them proactively as they happen. In conjunction with a logging solution, alerting is an essential tool that enables DevOps teams automate crucial elements of monitoring and maintaining app performance.

Alerting

PagerDuty
ServiceNow
VictorOps

Logging

Splunk
SumoLogic
Graylog
Loggly
Logentries

Rinse / Repeat

DevOps is continuous. There isn't a clear beginning or end to the DevOps Lifecycle. All of the stages of the DevOps lifecycle can potentially feed or lead to any other stage, resulting in a continuous cycle of rinse and repeat.

It all starts with embracing a DevOps culture and establishing a solid foundation of cloud and virtualization. Beyond that, the cycle of planning, designing, architecting, implementing, deploying, maintaining, and running apps is like a Mobius strip that continuously repeats. The continuous and multi-faceted aspects of the DevOps Lifecycle are what make it an effective and efficient approach to app development—constantly adapting and evolving.

Throughout the DevOps Lifecycle one of the most important elements is proactive monitoring. DevOps environments are simply too complex and fluid to manage with manual processes and most traditional approaches to IT infrastructure monitoring are incapable of keeping up with the rapid pace of DevOps. In order to rinse and repeat effectively organizations need a monitoring solution designed for DevOps.

Monitoring DevOps

There are a lot of platforms and tools available to facilitate a DevOps environment. One tool that stands out, though, in its ability to provide broad value across various stages of the DevOps lifecycle is a great application performance monitoring (APM) solution. A tool like AppDynamics APM provides your organization with valuable insight and gives you the ability to function more efficiently and make more intelligent decisions across all of the phases of the DevOps lifecycle.

AppDynamics APM enables you to detect potential issues before they impact the customer (or employee). The ability to safeguard and optimize application performance and address issues proactively helps organizations embrace the concept of fail-fast, fail-forward—driving innovation through rapid deployment without fear of the effects it will have on customer experience.

The AppDynamics platform also includes automation features that allow it to respond quickly and proactively to address issues that could impact application performance or the customer experience. When demand spikes and server resources are maxed out AppDynamics APM can automatically generate new server instances.

The integrated application analytics capabilities put actionable business intelligence at the fingertips of everyone. Data is displayed in a context appropriate for the audience, making it easy for both technical and non-technical individuals to gain valuable strategic insight. The AppDynamics war room feature also simplifies collaboration by ensuring that management, developers, and operations personnel all have access to the same information in real-time so effective decisions can be made to resolve issues.

AppDynamics APM gives you the insight and control you need to improve collaboration between developers, operations, and the business as a whole. It supports your continuous delivery strategy with real-time application data and automation of critical functions.